ДОДАТОК А ПРОГРАМНИЙ КОД

```
In [113]:
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.decomposition import TruncatedSVD
from sklearn.preprocessing import StandardScaler
from sklearn import decomposition, pipeline, metrics,
grid_search
In [114]:
def rmsle(h, y):
    """
    Compute the Root Mean Squared Log Error for hypthesis h
and targets y

    Args:
        h - numpy array containing predictions with shape
(n_samples, n_targets)
        y - numpy array containing targets with shape
(n_samples, n_targets)
    """
    return np.sqrt(np.square(np.log(h + 1) - np.log(y +
1)).mean())
In [115]:
weather = pd.read_csv('weather/weather_ready_to_use.csv')
weather['visit_date'] =
pd.to_datetime(weather['visit_date'],format= '%Y-%m-%d')
In [116]:
air_reserve = pd.read_csv('air_reserve.csv')
hpg_reserve = pd.read_csv('hpg_reserve.csv')
In [117]:
mean_day_wise = pd.read_csv('mean_day_wise.csv')
In [118]:
hpg_store_info = pd.read_csv('hpg_store_info.csv')
air_store_info = pd.read_csv('air_store_info.csv')
In [119]:
hpg_store_info.columns
Out[119]:
Index([u'hpg_store_id', u'hpg_genre_name', u'hpg_area_name',
u'latitude',
       u'longitude'],
      dtype='object')
In [120]:
```

```
n1 =
hpg_store_info.groupby('hpg_genre_name').agg('count')['hpg_sto
re_id'].reset_index()
n1.columns = ['hpg_genre_name','air_genre_name_c']

n2 =
hpg_store_info.groupby('hpg_area_name').agg('count')['hpg_stor
e_id'].reset_index()
n2.columns = ['hpg_area_name','air_area_name_c']
In [121]:
air_tf = list(air_store_info.apply(lambda x:'%s %s' %
(x['air_area_name'],x['air_genre_name']),axis=1))

tfv = TfidfVectorizer(min_df=3,  max_features=None,
            strip_accents='unicode',
analyzer='word',token_pattern=r'\w{1,}',
            ngram_range=(1, 5),
use_idf=1,smooth_idf=1,sublinear_tf=1,
            stop_words = 'english')
tfv.fit(air_tf)
Out[121]:
TfidfVectorizer(analyzer='word', binary=False,
decode_error=u'strict',
        dtype=<type 'numpy.int64'>, encoding=u'utf-8',
input=u'content',
        lowercase=True, max_df=1.0, max_features=None,
min_df=3,
        ngram_range=(1, 5), norm=u'l2', preprocessor=None,
smooth_idf=1,
        stop_words='english', strip_accents='unicode',
sublinear_tf=1,
        token_pattern='\\w{1,}', tokenizer=None, use_idf=1,
        vocabulary=None)
In [122]:
air_visit_data = pd.read_csv('air_visit_data.csv')
sample_submission = pd.read_csv('sample_submission.csv')
sample_submission['air_store_id'] =
sample_submission['id'].apply(lambda x:
x.split('_')[0]+str('_') +x.split('_')[1])
sample_submission['visit_date'] =
sample_submission['id'].apply(lambda x: x.split('_')[2])
sample_submission['visit_date'] =
pd.to_datetime(sample_submission['visit_date'],format=
'%Y-%m-%d %H:%M:%S')
In [123]:
air_visit_data = air_visit_data.merge(air_store_info,how =
'left',on= 'air_store_id')
sample_submission = sample_submission.merge(air_store_info,how
= 'left',on= 'air_store_id')
```

```
In [124]:
air_visit_data['id'] = air_visit_data['air_store_id'] +
str('_') + air_visit_data['visit_date']
new_train = pd.read_csv('new_train.csv')
air_visit_data = air_visit_data.merge(new_train,on='id',how=
'left')
air_visit_data = air_visit_data.drop('id',axis=1)
air_visit_data['visit_date'] =
pd.to_datetime(air_visit_data['visit_date'],format=
'%Y-%m-%d')
air_visit_data.visit_date.min()
Out[124]:
Timestamp('2016-01-01 00:00:00')
In [125]:
new_test =  pd.read_csv('new_test.csv')
sample_submission =  sample_submission.merge(new_test,how =
'left',on='id')
In [126]:
"""
def f(x):
    try:
        return x.split('-')[2]
    except:
        return -1

air_store_info['lol3'] = air_store_info.air_area_name.apply(f)
air_store_info['lol1'] =
air_store_info.air_area_name.apply(lambda x: x.split('-')[0])
air_store_info['lol2'] =
air_store_info.air_area_name.apply(lambda x: x.split('-')[1])
"""
Out[126]:
"\ndef f(x):\n    try:\n        return x.split('-')[2]\n
except:\n        return -1\n\nair_store_info['lol3'] =
air_store_info.air_area_name.apply(f)\nair_store_info['lol1']
= air_store_info.air_area_name.apply(lambda x: x.split('-
')[0])\nair_store_info['lol2'] =
air_store_info.air_area_name.apply(lambda x: x.split('-
')[1])\n"
In [127]:
air_store_info.columns
Out[127]:
Index([u'air_store_id', u'air_genre_name', u'air_area_name',
u'latitude',
       u'longitude'],
      dtype='object')
In [ ]:

In [128]:
```

```
air_reserve['visit_datetime'] =
pd.to_datetime(air_reserve['visit_datetime'],format=
'%Y-%m-%d %H:%M:%S')
air_reserve['visit_date'] =
air_reserve.visit_datetime.apply(lambda x: str(x).split('
')[0])
air_reserve['visit_date'] =
pd.to_datetime(air_reserve['visit_date'],format=
'%Y-%m-%d %H:%M:%S')
In [129]:
hpg_reserve['visit_datetime'] =
pd.to_datetime(hpg_reserve['visit_datetime'],format=
'%Y-%m-%d %H:%M:%S')
hpg_reserve['visit_date'] =
hpg_reserve.visit_datetime.apply(lambda x: str(x).split('
')[0])
hpg_reserve['visit_date'] =
pd.to_datetime(hpg_reserve['visit_date'],format=
'%Y-%m-%d %H:%M:%S')
In [ ]:

In [130]:
for i in ['reserve_datetime','reserve_visitors']:
        k =
air_reserve[[i,'visit_date','air_store_id']].groupby(['visit_d
ate','air_store_id'])[i].apply(lambda x:
x.tolist()).reset_index()
        name = i + 'list'
        if i == 'reserve_datetime':
            k1  = k.copy()
        else:
            k1[name] = k[i].copy()
        print i
reserve_datetime
reserve_visitors
In [131]:
air_visit_data =
air_visit_data.merge(k1,on=['air_store_id','visit_date'],how =
'left')
sample_submission =
sample_submission.merge(k1,on=['air_store_id','visit_date'],ho
w = 'left')
In [ ]:

In [ ]:

In [132]:
air_visit_data['visit_date_month']
=air_visit_data.visit_date.dt.month
```

```
air_visit_data['visit_date_dayofw']
=air_visit_data.visit_date.dt.dayofweek
air_visit_data['visit_date_year']
=air_visit_data.visit_date.dt.year
air_visit_data['visit_date_dayofm']
=air_visit_data.visit_date.dt.day
air_visit_data['weekofyear']
=air_visit_data.visit_date.dt.weekofyear
sample_submission['visit_date_month']
=sample_submission.visit_date.dt.month
sample_submission['visit_date_dayofw']
=sample_submission.visit_date.dt.dayofweek
sample_submission['visit_date_year']
=sample_submission.visit_date.dt.year
sample_submission['visit_date_dayofm']
=sample_submission.visit_date.dt.day
sample_submission['weekofyear']
=sample_submission.visit_date.dt.weekofyear
air_visit_data.loc[air_visit_data.weekofyear==53,'weekofyear']
=0
sample_submission.loc[sample_submission.weekofyear==53,'weekof
year'] =0
In [133]:
total_air_ids = list(air_store_info.air_store_id.unique())

df_total = pd.concat((air_visit_data,sample_submission))
df_total =
df_total[['weekofyear','visit_date_year','air_store_id']]
In [134]:
df_total =
df_total.sort_values(['visit_date_year','weekofyear'])
weekofyear = list(df_total['weekofyear'].unique())
year = list(df_total['visit_date_year'].unique())
In [ ]:

In [135]:
week_open_restro = []
for i in year:
    for j in weekofyear:
        #print
i,j,len(list(df_total.loc[(df_total.visit_date_year ==i) &
(df_total.weekofyear ==j),'air_store_id'].unique()))
        l= len(list(df_total.loc[(df_total.visit_date_year
==i) & (df_total.weekofyear ==j),'air_store_id'].unique()))
        if (i==2017) & (j>22):
            break
        week_open_restro.append([i,j,l])
In [136]:
df_open_restro = pd.DataFrame(week_open_restro)
```

```
df_open_restro.columns =
['visit_date_year','weekofyear','no_open_restro']
In [137]:
def func1(x):
    try:
        if pd.isnull(x):
            return 0
        else:
            return len(x)

    except:
        return len(x)
def func(x):
    try:
        if pd.isnull(x):
            return -1
        else:
            return sum(x)
    except:
        return sum(x)

air_visit_data['total_reserve']=
air_visit_data['reserve_visitorslist'].apply(func)
air_visit_data['numb_total_reserve'] =
air_visit_data['reserve_visitorslist'].apply(func1)

sample_submission['total_reserve']=
sample_submission['reserve_visitorslist'].apply(func)
sample_submission['numb_total_reserve'] =
sample_submission['reserve_visitorslist'].apply(func1)
In [138]:
k = [i for i in air_visit_data.columns if i in
sample_submission.columns]
train = air_visit_data.copy()
test = sample_submission.copy()
In [139]:
train.columns
Out[139]:
Index([u'air_store_id', u'visit_date', u'visitors',
u'air_genre_name',
       u'air_area_name', u'latitude', u'longitude',
u'holiday_eve',
       u'non_working', u'genre_in_area', u'total_r_in_area',
       u'reserve_visitors', u'reserve_-12_h',
u'reserve_12_37_h',
       u'reserve_37_59_h', u'reserve_59_85_h',
u'reserve_85+_h',
       u'visitors_mean', u'visitors_median', u'visitors_max',
u'visitors_min',
```

```
       u'reserve_datetime', u'reserve_visitorslist',
u'visit_date_month',
       u'visit_date_dayofw', u'visit_date_year',
u'visit_date_dayofm',
       u'weekofyear', u'total_reserve',
u'numb_total_reserve'],
      dtype='object')
In [140]:
"""
l =  pd.to_datetime('2017-02-1',format='%Y-%m-%d')
train2 = train.loc[(train.visit_date>=l)]"""
Out[140]:
"\nl =  pd.to_datetime('2017-02-1',format='%Y-%m-%d')\ntrain2
= train.loc[(train.visit_date>=l)]"
In [141]:
train.visit_date.min(),train.visit_date.max()
Out[141]:
(Timestamp('2016-01-01 00:00:00'), Timestamp('2017-04-22
00:00:00'))
In [ ]:

In [142]:
train1 = train.loc[(train.visit_date_year>=2017)].copy()
k1 =
train1[['visitors','air_store_id']].groupby('air_store_id').ag
g('mean').reset_index()
k1.columns = ['air_store_id','mean_visitors']
k2 =
train1[['visitors','air_store_id']].groupby('air_store_id').ag
g('median').reset_index()
k2.columns = ['air_store_id','median_visitors']
In [143]:
k4 =
train[['visitors','visit_date_month']].groupby(['visit_date_mo
nth']).agg('mean').reset_index()
k4.columns = ['visit_date_month','mean_visitors2']

k5 =
train1[['visitors','air_store_id','visit_date_dayofw']].groupb
y(['air_store_id','visit_date_dayofw']).agg('mean').reset_inde
x()
k5.columns = ['air_store_id','visit_date_dayofw','xxx']

k6 =
train1[['visitors','visit_date_dayofw']].groupby(['visit_date_
dayofw']).agg('mean').reset_index()
k6.columns = ['visit_date_dayofw','mean_visitors4']
```

```
k7 =
train[['visitors','visit_date_month']].groupby(['visit_date_mo
nth']).agg('median').reset_index()
k7.columns = ['visit_date_month','median_visitors2']

k8 =
train1[['visitors','air_store_id','visit_date_dayofw']].groupb
y(['air_store_id','visit_date_dayofw']).agg('median').reset_in
dex()
k8.columns = ['air_store_id','visit_date_dayofw','yyy']

k9 =
train1[['visitors','visit_date_dayofw']].groupby(['visit_date_
dayofw']).agg('median').reset_index()
k9.columns = ['visit_date_dayofw','median_visitors4']


"""
total =
pd.concat((train[['air_store_id','air_genre_name','air_area_na
me']],test[['air_store_id','air_genre_name','air_area_name']])
)


k10 =
total.groupby(['air_genre_name']).agg('count')['air_store_id']
.reset_index()
k10.columns = ['air_genre_name','count1']
k11 =
total.groupby(['air_area_name']).agg('count')['air_store_id'].
reset_index()
k11.columns = ['air_area_name','count2']

k12 =
total.groupby(['air_store_id']).agg('count')['air_area_name'].
reset_index()
k12.columns = ['air_store_id','count2']

"""

k10 =
train[['visitors','air_store_id']].groupby('air_store_id').agg
('mean').reset_index()
k10.columns = ['air_store_id','mean_visitors_f']
k11 =
train[['visitors','air_store_id','visit_date_dayofw']].groupby
(['air_store_id','visit_date_dayofw']).agg('mean').reset_index
()
```

```
k11.columns =
['air_store_id','visit_date_dayofw','mean_visitors3_f']
k12 =
train[['visitors','visit_date_dayofw']].groupby(['visit_date_d
ayofw']).agg('mean').reset_index()
k12.columns = ['visit_date_dayofw','mean_visitors4_f']

k13 =
train[['visitors','weekofyear']].groupby(['weekofyear']).agg('
mean').reset_index()
k13.columns = ['weekofyear','mean_visitors2_weekofyear']
k14 =
train[['visitors','weekofyear']].groupby(['weekofyear']).agg('
median').reset_index()
k14.columns = ['weekofyear','median_visitors2_weekofyear']

k15 =
train[['visitors','air_store_id','visit_date_month']].groupby(
['air_store_id','visit_date_month']).agg('mean').reset_index()

#k12 =
train[['visitors','air_genre_name','air_area_name']].groupby([
'air_genre_name','air_area_name']).agg('mean').reset_index()
#k12.columns =
['air_genre_name','air_area_name','mean_air_area_name_visitor'
]
In [144]:
"""
k15.columns = ['air_store_id','visit_date_month','xxx']
train =
train.merge(k15,on=['air_store_id','visit_date_month'],how='le
ft')
test = test.merge(k15,on=
['air_store_id','visit_date_month'],how='left')
k1.columns = ['air_store_id', 'yyy']
test= test.merge(k1,on = 'air_store_id',how = 'left')
test.loc[pd.isnull(test.xxx),'xxx'] =
test.loc[pd.isnull(test.xxx),'yyy'].copy()
test = test.drop('yyy',axis=1)
"""
Out[144]:
"\nk15.columns =
['air_store_id','visit_date_month','xxx']\ntrain =
train.merge(k15,on=['air_store_id','visit_date_month'],how='le
ft')\ntest = test.merge(k15,on=
['air_store_id','visit_date_month'],how='left')\nk1.columns =
['air_store_id', 'yyy']\ntest= test.merge(k1,on =
'air_store_id',how =
'left')\ntest.loc[pd.isnull(test.xxx),'xxx'] =
```

```
test.loc[pd.isnull(test.xxx),'yyy'].copy()\ntest =
test.drop('yyy',axis=1)\n"
In [ ]:
In [145]:
train = air_visit_data.copy()
test = sample_submission.copy()
train = train[k]
test = test[k]

train = train.merge(k1,on='air_store_id',how='left')
test = test.merge(k1,on='air_store_id',how='left')
train = train.merge(k2,on='air_store_id',how='left')
test = test.merge(k2,on='air_store_id',how='left')
#train =
train.merge(k3,on=['air_store_id','visit_date_month'],how='lef
t')
#test = test.merge(k3,on=
['air_store_id','visit_date_month'],how='left')

train = train.merge(k4,on=['visit_date_month'],how='left')
test = test.merge(k4,on= ['visit_date_month'],how='left')
#train =
train.merge(k5,on=['air_store_id','visit_date_dayofw'],how='le
ft')
#test = test.merge(k5,on=
['air_store_id','visit_date_dayofw'],how='left')
train = train.merge(k6,on=['visit_date_dayofw'],how='left')
test = test.merge(k6,on= ['visit_date_dayofw'],how='left')


train = train.merge(k7,on=['visit_date_month'],how='left')
test = test.merge(k7,on= ['visit_date_month'],how='left')
#train =
train.merge(k8,on=['air_store_id','visit_date_dayofw'],how='le
ft')
#test = test.merge(k8,on=
['air_store_id','visit_date_dayofw'],how='left')
train = train.merge(k9,on=['visit_date_dayofw'],how='left')
test = test.merge(k9,on= ['visit_date_dayofw'],how='left')

train = train.merge(k10,on=['air_store_id'],how='left')
test = test.merge(k10,on= ['air_store_id'],how='left')
train =
train.merge(k11,on=['air_store_id','visit_date_dayofw'],how='l
eft')
test = test.merge(k11,on=
['air_store_id','visit_date_dayofw'],how='left')
train = train.merge(k12,on=['visit_date_dayofw'],how='left')
test = test.merge(k12,on= ['visit_date_dayofw'],how='left')
```

```
train =
train.merge(df_open_restro,on=['visit_date_year','weekofyear']
,how='left')
test = test.merge(df_open_restro,on=
['visit_date_year','weekofyear'],how='left')



#train = train.merge(k13,on=['weekofyear'],how='left')
#test = test.merge(k13,on= ['weekofyear'],how='left')
#train = train.merge(k14,on=['weekofyear'],how='left')
#test = test.merge(k14,on= ['weekofyear'],how='left')



"""
n1 =
air_store_info.groupby(['air_genre_name','air_area_name']).agg
('count')['air_store_id'].reset_index()
n1.columns =
['air_genre_name','air_area_name','air_genre_name_c']


n2 =
n1.groupby('air_genre_name').agg('count')['air_area_name'].res
et_index()
n2.columns = ['air_genre_name','air_area_name_unique']

n3 =
n1.groupby('air_area_name').agg('count')['air_genre_name'].res
et_index()
n3.columns = ['air_area_name','air_genre_name_unique']



train =
train.merge(k14,on=['air_store_id','visit_date_dayofw'],how='l
eft')
test = test.merge(k14,on=
['air_store_id','visit_date_dayofw'],how='left')
train = train.merge(k15,on=['visit_date_dayofw'],how='left')
test = test.merge(k15,on= ['visit_date_dayofw'],how='left')
"""
y = train.visitors.values
ids = sample_submission['id'].values
In [146]:
#test.loc[pd.isnull(test.xxx),['yyy','xxx']]
```

```
In [147]:
date_info = pd.read_csv('date_info.csv')
date_info['calendar_date']  =
pd.to_datetime(date_info['calendar_date'],format= '%Y-%m-%d')
date_info.rename(columns =
{'calendar_date':'visit_date'},inplace = True)

print('holidays at weekends are not special, right?')
wkend_holidays = date_info.apply((lambda
x:(x.day_of_week=='Sunday' or x.day_of_week=='Saturday') and
x.holiday_flg==1), axis=1)
date_info.loc[wkend_holidays, 'holiday_flg'] = 0
date_info['weight'] = ((date_info.index + 1.0) /
len(date_info)) ** 5.0


train = train.merge(date_info,on='visit_date',how='left')
test = test.merge(date_info,on='visit_date',how='left')
holidays at weekends are not special, right?
In [148]:
"""
k1 = train.loc[train.holiday_flg ==
1].groupby(['air_store_id','visit_date_dayofw'])['visitors'].a
gg('mean').reset_index()
k2 = train.loc[train.holiday_flg !=
1].groupby(['air_store_id','visit_date_dayofw'])['visitors'].a
gg('mean').reset_index()
k1.columns =
['air_store_id','visit_date_dayofw','holiday_flg_1']
k2.columns =
['air_store_id','visit_date_dayofw','holiday_flg_1']

k =
k1.merge(k2,on=['air_store_id','visit_date_dayofw'],how='left'
)
train =
train.merge(k,how='left',on=['air_store_id','visit_date_dayofw
'])
test =
test.merge(k,how='left',on=['air_store_id','visit_date_dayofw'
])
k1 = train.loc[train.holiday_flg ==
1].groupby(['air_area_name','visit_date_dayofw'])['visitors'].
agg('mean').reset_index()
k2 = train.loc[train.holiday_flg !=
1].groupby(['air_area_name','visit_date_dayofw'])['visitors'].
agg('mean').reset_index()
k1.columns =
['air_area_name','visit_date_dayofw','holiday_flg_1']
```

```
k2.columns =
['air_area_name','visit_date_dayofw','holiday_flg_1']

k =
k1.merge(k2,on=['air_area_name','visit_date_dayofw'],how='left
')
train =
train.merge(k,how='left',on=['air_area_name','visit_date_dayof
w'])
test =
test.merge(k,how='left',on=['air_area_name','visit_date_dayofw
'])
"""
Out[148]:
"\nk1 = train.loc[train.holiday_flg ==
1].groupby(['air_store_id','visit_date_dayofw'])['visitors'].a
gg('mean').reset_index()\nk2 = train.loc[train.holiday_flg !=
1].groupby(['air_store_id','visit_date_dayofw'])['visitors'].a
gg('mean').reset_index()\nk1.columns =
['air_store_id','visit_date_dayofw','holiday_flg_1']\nk2.colum
ns = ['air_store_id','visit_date_dayofw','holiday_flg_1']\n\nk
=
k1.merge(k2,on=['air_store_id','visit_date_dayofw'],how='left'
)\ntrain =
train.merge(k,how='left',on=['air_store_id','visit_date_dayofw
'])\ntest =
test.merge(k,how='left',on=['air_store_id','visit_date_dayofw'
])\nk1 = train.loc[train.holiday_flg ==
1].groupby(['air_area_name','visit_date_dayofw'])['visitors'].
agg('mean').reset_index()\nk2 = train.loc[train.holiday_flg !=
1].groupby(['air_area_name','visit_date_dayofw'])['visitors'].
agg('mean').reset_index()\nk1.columns =
['air_area_name','visit_date_dayofw','holiday_flg_1']\nk2.colu
mns =
['air_area_name','visit_date_dayofw','holiday_flg_1']\n\nk =
k1.merge(k2,on=['air_area_name','visit_date_dayofw'],how='left
')\ntrain =
train.merge(k,how='left',on=['air_area_name','visit_date_dayof
w'])\ntest =
test.merge(k,how='left',on=['air_area_name','visit_date_dayofw
'])\n"
```

відомості кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	72 с.
	Інші документи	
2.	Презентаційні матеріали	

		Прізвище та ініціали.	Підп.	Дата			
Розробив		ЯковлєвЯ.В.			Дослідження методів тематичного моделювання корпусів текстів на прикладі контенту сайту	Шифр групи	Код напр./спец.
Перевірив		Терзіян В.Я.				СШІм-19-1	122
Н.контр.		Малєєва І.А.				ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.					